Cluster Computing
https://doi.org/10.1007/s10586-024-04399-2

Multi-modal Battle Royale optimizer

K. Dilşad Çiçek[1] · Taymaz Akan[2,3] · Oğuz Bayat[4]

Received: 27 October 2023 / Revised: 27 February 2024 / Accepted: 28 February 2024

Abstract

Multimodal optimization poses a challenging problem in the field of optimization as it entails the discovery of multiple local and global optima, unlike unimodal optimization, which seeks a single global solution. In recent years, the significance of addressing multimodal optimization challenges has grown due to the real-world complexity of many problems. While numerous optimization methods are available for unimodal problems, multimodal optimization techniques have garnered increased attention. However, these approaches often grapple with a common issue: the determination of the niching parameter, necessitating prior knowledge of the problem space. This paper introduces a novel multimodal optimization approach that circumvents the need for prior problem space knowledge and avoids the challenge of predefining the niching parameter. Building upon the Battle Royal Optimization (BRO) algorithm, this extended version formulates a multimodal solution by utilizing Coulomb's law to identify suitable neighbors. The incorporation of Coulomb's law serves the dual purpose of identifying potential local and global optima based on fitness values and establishing optimal distances from solution candidates. A comparison study was done between the MBRO and seven well-known multimodal optimization algorithms using 14 benchmark problems from the CEC 2013 and CEC 2015 competitions to see how well it worked. The experimental results underscore MBRO's proficiency in successfully identifying most, if not all, local and global optima, positioning it as a superior solution when compared to its competitors.

Keywords Battle Royale optimization · Multi-modal optimization · Local search

1 Introduction

Optimization involves finding the optimum solution among a set of potential solutions in a particular optimization problem. Most metaheuristic optimization algorithms are designed to solve unimodal optimization problems [1]. However, many real-world problems have more than one solution due to their nature. However, due to the fact that they employ a strategy of escaping from local optima, unimodal algorithms are unable to locate local optima. Herein, multimodal algorithms provide decision-makers with alternative solutions. Multi-modal optimization algorithms are designed to find multiple optima (both global and local) simultaneously within a given problem space. Unimodal optimization algorithms need to preserve diversity when adapted to multi-modal optimization algorithms. Niching strategies have been widely applied to multimodal optimization problems to divide the population into sub-populations [2]. Niching methods can be divided into many categories: Fitness Sharing [3–8], Crowding [9, 10], Clearing [11], Speciation [12] and more [13].

Crowding is one of the oldest and simplest techniques used to solve multimodal optimization problems. By using the distance between similar individuals in the population, De Jong's crowding method [14] maintains population diversity. The algorithm compares a generated solution with some random solutions from the current population. If the generated solution is better, it will be replaced with the most similar solution. The biggest advantage of crowding is its simplicity. However, the replacement error is the main drawback of

✉ Taymaz Akan
taymaz.farshi@gmail.com

1 Graduate School of Science and Engineering, Altinbas University, Istanbul, Turkey

2 Department of Medicine, Louisiana State University Health Sciences Center, Shreveport, LA, USA

3 Department of Software Engineering, Faculty of Engineering, Topkapı University, Istanbul, Turkey

4 Faculty of Engineering and Architecture, Beykoz University, Istanbul, Turkey

Published online: 14 April 2024

crowding. For instance, Mahfoud [10] explains how stochastic "replacement errors" prevent the fundamental crowding algorithm from keeping more than two peaks in a multimodal fitness landscape. In order to find and keep track of multiple peaks in the presence of a multimodal function, Mahfoud details an enhanced crowding algorithm that is called deterministic crowding. Each generation of deterministic crowding pairs all members of the population at random, resulting in n = 2 pairs, where n is the population size. Crossover, in addition to mutations and other genetic operators, can produce offspring from each such parent pair. Both children then have to compete with their parents to join the population. The deterministic crowding method shows a strong ability to solve some multi-peak optimization problems and preserve niches. However, the Deterministic Crowding replacement policy results in the loss of low availability niches. For this purpose, [15] proposes the probability crowding technique, allowing the offspring to be changed according to the fitness values of similar parents.

The fitness sharing approach proposed by Holland [16] and later developed by Goldberg and Richardson [3] divides the different populations into subgroups by considering the similar characteristics of individuals in the population. The advantage of sharing is that it creates a subpopulation sustainably using niching. Increases population diversity by increasing searches in unexplored areas of shared space. The downside is that it needs a specific parameter. Specifying this parameter means that it requires prior knowledge of how far from the solution it is. However, this is not possible for real-world problems [7].

Clearing is another widely used niching method [11]. Unlike fitness sharing, clearing keeps only the best individuals in a niching, eliminating the bad ones. The algorithm first sorts in descending order by considering the fitness value in the population. Then, after choosing a solution from the top, it eliminates individuals with worse fit than the chosen one. Clearing preserves diversity among selected individuals. Like the sharing operation in clearing, it needs a user-specified σ_{clear} parameter. The advantage of the clearing process over the sharing process is that it is simple. It also handles the best elements across generations. However, convergence is slow in clearing and local solutions are not effective.

Species method is widely used in multimodal optimization [11, 12, 17]. This method depends on the radius parameter to the central boundary of a type measuring Euclidean distance. All individuals within the specified radius are calculated from the same species. In this way, the entire population is divided into different groups based on their similarities. The main advantage of species method is the ability to maintain high diversity over generations. The disadvantage is that it needs the radius parameter, like other niche-based approaches.

The majority of current niching methods necessitate the use of predetermined niching parameters or additional knowledge regarding the problem domain. However, there have been efforts to develop multi-modal optimization algorithms without niche parameters or that are more adaptive and do not require extensive parameter tuning or domain-specific knowledge. These algorithms aim to automatically adjust their niching parameters and adapt to the characteristics of the problem at hand. They use mechanisms such as self-adaptive niching radii or dynamic clustering methods. Some of the suggested Multimodal optimization algorithms without the need for any niche parameters are as follows.

Li [1] adapted a fitness distance ratio strategy to the PSO to extend the PSO to a multimodal optimization form. The FER determines the fitness difference between the particles and the rest of the swarm by calculating the Euclidean distance. By maximizing the ratio of fitness and distance, it directs the movement of one particle to other particles not only with respect to fitness value but also with respect to distance.

EPSO (Electrostatic PSO) is a PSO-based algorithm based on electrostatic interaction without the need for any additional parameters [18]. Here, the electrostatic interaction between particles was computed by employing Coulomb's law. Most of the particles converged to the local and global optima through the application of Coulomb's law. To achieve an appropriate distance from the particle, particles attempted to move toward a point where the cost function value was also appropriate. Moreover, [19] adapted a local search to some PSO-based multimodal algorithms to enhance a convergence.

Apart from Particle Swarm Optimization (PSO)-based multimodal algorithms, there are various optimization algorithms that have been extended or adapted to handle multimodal optimization problems. These adaptations aim to enable the exploration of multiple optima in the search space. Here are some examples of optimization algorithms that have been extended for multimodal optimization: Niche Gravitational Search Algorithm (NGSA) [20], Multimodal Firefly Algorithm [20], Multimodal Flower Pollination Algorithm [21],Multimodal Animal Migration Optimization Algorithm (AMO) [22], Multimodal Bacterial Foraging Optimization (MBF0) [23], Multimodal Butterfly Optimization Using Fitness-Distance Balance [24] and etc.

2 Battle Royale optimization algorithm

Battle Royale Optimization (BRO) Algorithm was introduced by Taymaz in 2020 as a game-based optimization algorithm inspired by the last survival strategy in a challenging environment [25].

In these types of fighting games, players (soldiers) fight against each other in the context of competition. The goal for each player is to first stay alive and then kill as many other players as possible. While playing, if a player keeps getting hurt for a certain amount of time, it will respawn in a random part of the game field. Like in Battle Royale games, the BRO algorithm spreads out the first possible solutions randomly in the problem space. Afterward, each answer would be compared to its nearest neighbor. The solution with the higher fitness value would be named the winner, while the other solution would be named the loser. There is a parameter in each candidate solution that stores the damage (lose) level of each solution. This parameter would go up after each damage. If a solution is damaged over and over for a certain amount of time (depending on the problem to be solved), it will be moved around according to Eq. (1) the damage level will be reset to zero. If the amount of damage is below the threshold, then Eq. (2) will be used to effect reallocation.

$$x_{dam,d} = r(ub_d - lb_d) + lb_d \tag{1}$$

$$x_{dam,d} = x_{dam,d} + r\left(x_{best,d} - x_{dam,d}\right) \tag{2}$$

In these equations, r is a uniformly distributed random number in the interval [0,1], and $x_{dam,d}$ and $x_{best,d}$ represent the locations of the damaged and best-known solutions in dimension d, respectively. Lower and upper bounds on problem space dimension d are denoted by ub_d and lb_d, respectively.

The essential feature of the algorithm is that the search space gets smaller and closer to the optimal solution at each Δ iteration by $\Delta = \Delta + \lceil \frac{\Delta}{2} \rceil$ if iteration $\geq \Delta$. If iteration is increased to a value of Δ, the safety zone will decrease in size. Δ has been set up by default to be $\frac{MaxCicle}{\text{round}(\log_{10}(MaxCicle))}$, where $MaxCicle$ is the maximum number of iterations. The idea behind this space limit is to move all possible solutions toward the one that might be the best. To support elitism, keep in mind that the best solution from each round is saved. Using Eq. (3), where $SD(\overline{x_d})$ is the population standard deviation in dimension d, we can reduce the size of the problem space.

$$\begin{aligned} lb_d &= x_{best,d} - SD(\overline{x_d}) \\ ub_d &= x_{best,d} + SD(\overline{x_d}) \end{aligned} \tag{3}$$

3 Multi-modal Battle Royale optimization algorithm (MBRO)

This proposed method presents an extended version of BRO algorithm, one that can find multiple local and global optima. This section will describe the Multi-Modal BRO (MBRO) algorithm. The proposed method relies solely on the parameters of the original BRO algorithm. The determination of the appropriate niching radius is the primary challenge that the majority of the existing methods must overcome. However, this method gets rid of the need for domain knowledge by not using a niching parameter. The complexity of the proposed multi-modal algorithm is the same as the original version as it does not necessitate any extra parameter. Like other multimodal optimization methods, the proposed MBRO needs to be changed in certain ways so that it can find more than one optimal solution.

As mentioned before, each solution is competing with another that is close in radius. This means that a solution with a more advantageous position has a greater chance of success. As a result, everyone is trying to improve their position in order to increase their chances of survival. Among the corresponding solution and its nearest neighbor, the solution which located in the better positions therefore cause increasing the damage score to another one. In order to find the nearest neighbor, the BRO algorithm uses Euclidean distance. The first modification is performed on Coulomb's law (Eq. 4) which provides attractive or repulsive interaction between solutions.

$$|F| = K\frac{Q_1 . Q_2}{r^2} \tag{4}$$

where F is the magnitude of the electrostatic force, Q_1 and Q_2 are magnitudes of the charges, and finally K is is the Coulomb constant($K \approx \frac{1}{4\pi\varepsilon_0}$).Therefore, instead of the Euclidean distance the electrostatic interactions between the i^{th} individual and the rest of the population are computed and saved in the vector$F$. By adopting this strategy, the indices of neighbors for the i^{th} individual can be determined through the following equation proposed in [1]:

$$F_{i,j} = \alpha\frac{f(x_i)f\left(x_j\right)}{r^2} \tag{5}$$

Here $f(x_i)$ and $f(x_j)$ are the objective value of i^{th} and j^{th} solutions, respectively. α is considered as Coulomb constant taken to be 1.

In this case, the index of the solution to be compared with the i^{th} solution is calculated per argmax(f_i(j)). Therefore, instead of comparing a solution with the nearest neighbor, it compares with a solution that provides the largest electrostatic force.

Figure 1 appears to be a visual aid from illustrating the Coulomb's law integration, this figure illustrates the concept of computing an F vector in a two-dimensional search space. It's used to represent a set of solutions (labeled A, B, C, D) that have been evaluated by a certain function f, and the distances between these points are also shown. Each point (A, B, C, D) represents a possible solution in the search space. The potential for improving "A" can be realized by moving A towards “fittest-and-closest” neighbors, which can be determined by calculating the F value for the individual particle. The function f has been applied to each point, yielding values $f(A) = 3$, $f(B) = 2$, $f(C) = 6$, and $f(D) = 10$. The distances between points A and the others are given by $\|A - B\| = 3$, $\|A - C\| = 6$, and $\|A - D\| = 8$. The F values, such as $F_{A,B}$, $F_{A,C}$ and $F_{A,D}$, are calculated using a Eq. 5, Coulomb’s law, to take into account both the objective values and the distances between solutions. These F values are used to determine the “fittest-and-closest” neighbor for solution A. According to F values, the “fittest-and-closest” neighbor for A is B.

When a solution’s damage level does not exceed a predetermined threshold, the soldier will execute a standard “run-and-tumble” motion in order to enhance the accuracy of the local search and improve the overall search capability. In this manner, solutions move about their environment in order to find a better position. A combination of forward stepping and random movement characterizes the individual’s path to a more advantageous position. Due to this mechanism, a solution moves randomly in a certain direction. If the solution is in a better spot, the random movement will continue from the new spot. If not, the solution will try to tumble in a new, random direction. This movement will only go on until a certain value is reached T = 5. In each step, the position of the solution is changed based on:

$$x_i = x_i + \lambda\Delta, \tag{6}$$

where Δ indicates the size of step and calculated per $\Delta = (ub - lb) \times rand[0, 1] + lb$. Through trial and error, it was found that the right values for lb (lower bound) and ub (upper bound) are -0.8 and 0.8, respectively. Moreover, λ is a control parameter for the step size.

On the other hand, if a solution’s damage level exceeds a predetermined threshold, the solution will move toward the winning solution using Eq. 7. As the algorithm gets closer to the end of the generation, the step size should be decreased so that the exploration and exploitation are in a more even balance. The following formula is used to compute w:

$$w = w_{max} - \left(\frac{w_{max} - w_{min}}{G}\right) \times G \tag{7}$$

where G is the maximum generation value, and $w_{max} = 0.4$ and $w_{min} = 0.0001$ are the lower and upper bounds of w, correspondingly. Hence,

$$x_{dam,d} = x_{dam,d} + wr\left(x_{best,d} - x_{dam,d}\right) \tag{8}$$

The last thing to notice is that the shrinking step is not performed as the aim is not to focus on a single optimum solution.

The computational complexity of the proposed method is contingent upon the population size and maximum number of iterations, in addition to the problem’s dimensions. In order determine the Euclidean distance between any two solutions, it is necessary to compare each solution with the others. Considering the population size n, the computational complexity for calculating the aforementioned distance is $O(n^2)$. Therefore, when the number of iterations m is considered, the computational complexity of BRO equals $O(n^3)$.

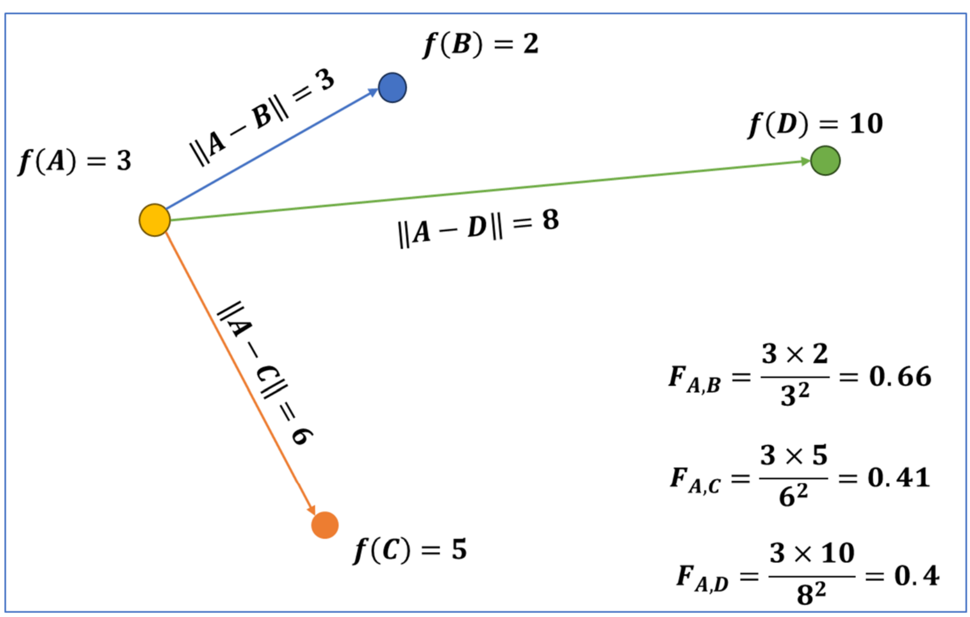

Fig. 1 An example of computing F vector in in a 2-dimensional search space. The population size is three. According to F values, the best neighbor for A is B. Maximization is assumed

4 Experimental results and performance evaluations

The experiments were executed utilizing MATLAB R2019 on a Windows 10 system equipped with a 32 GB RAM and a Core i7-7700HQ 2.80 GHz processor. This section contains a summary of the numerical experiments that were conducted using the algorithm that we have proposed. By contrasting MBRO with several multi-modal techniques, we demonstrate its efficacy in locating and sustaining both local and global optima. We provide a description of the optimization test functions and the performance criteria prior to presenting the results.

4.1 Test function

MBRO and its competitors (CFA [20], EPSO [18], FERPSO [6], LSPSO [19], NGSA [37], MFPA [39], MAMO, and [22]) have undergone testing on a set of 14 widely employed multi-modal benchmark functions from CEC 2013 and CEC 2015, as referenced in [26] [27]. The different features of these functions are listed in Table 1. Eight multi-modal functions (F1–F8) and two composition functions (F9 and F10) are shown in Table 1. A number of basic Grienwank, Weierstrass, and Sphere functions are used to make F9. The F10 function is made up of Rastrigin, Grienwank, Weierstrass, and Sphere functions. These equations are used to figure out the composite functions:

$$\omega_i = \exp\left(-\frac{\sum_{k=1}^{D}(x_k - o_{ik})^2}{2D\sigma_i^2}\right) \tag{9}$$

$$\omega_i = \begin{cases} \omega_i \ \omega_i = \max(\omega_i) \\ \omega_i\left(1 - max(\omega_i)^{10}\right) \ otherwise \end{cases} \tag{10}$$

In the given cases F8 and F9, the value of σ_i is established as 1. F9 represents a value of zero for both $bias_i$ and f_{bias}. Furthermore, considering i as $\{1, 2, \ldots, n\}$, o_i and M_i are the newly shifted optimums of the identity matrix and each f_i, respectively. Moreover, F11–F14 are the expanded scalable functions from CEC 2015.

The performance of the algorithm was evaluated using the following test functions.

4.2 Performance criteria

In this section, four multi-modal optimization metrics are presented. In 25 iterations, these criteria were evaluated in order to determine the effectiveness of the proposed method. These criteria specify both the quantity and quality of the obtained optima. The following criteria assessed algorithm performance.

1. Success Rates (SR); It is the value that shows the successful run percentage of the known optimal.

When calculating the success rate; the formula used is as follows.

$$SR = \frac{NSR}{NR} \tag{11}$$

To calculate the success rate, another parameter known as the level of accuracy must be defined. The level of accuracy is usually chosen within the range of [0, 1].

2. Average number of optima found: This metric calculates the average number of solutions obtained over the number of runs. When the success rate value for a particular difficult problem reaches zero, division by zero may take place, resulting in an undefinable success performance.
3. Success performance (SP): this criterion evaluates the probability of a high success rate with a lower number of function evaluations. SP is defined as

$$Success\ Performance = \frac{\sum_{i=1}^{N_R} N_{FE}}{N_R \times SR} \tag{12}$$

4. Maximum Peak Ratio (MPR): Inspired by the work of Miller and Shaw [6] it aims to find the local optima in the population to measure its optimum quality. MPR is used to test the quality of the found optimum calculated according to Equation.

$$\mathrm{MPR} = \frac{1}{N_R}\sum_{r=1}^{N_R}\left(\frac{\sum_{i=1}^{N_{OP_r}} f_i}{\sum_{i=1}^{N_{AP}} F_i}\right) \tag{13}$$

4.3 Experiments and results

The performance of MBRO has been compared using seven multimodal algorithms and fourteen test functions. The compared test functions are shown in Table 4. Two independent of the problem's dimensions variables that influence the computational complexity of a given algorithm are the population size and the maximum number of iterations. A comprehensive evaluation of each solution is necessary in order to transform the original form of each implemented algorithm into a multi-modal form. $O(n^2)$ is the complexity of calculating the Euclidean distance for every solution, where n represents the size of the population. Because each solution must be compared to every other in order to determine its distance from them, this is the case. Consequently, for any given number of iterations m, the computational complexity of each method is $O(n^3)$.

Table 1 Test functions

Function name	Equation	Search range	Number of local/global
F1: Ackley	$-a\exp\left(-b\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right)-\exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(cx_i)\right)+a+\exp(1)$	$-5\le x_i\le 5$	120/1
F2: Six Hump	$\left(4-2.1x_1^2+\frac{x_1^4}{4}\right)x_1^2+x_1x_2+\left(-4+4x_2^2\right)x_2^2$	$-1.9\le x_i\le 1.9$ $-1.1\le x_i\le 1.1$	4/2
F3: Decreasing Maxima	$\exp\left(-2\log(2)\left(\frac{x-0.08}{0.854}\right)^2\right)sin^6\left(5\pi\left(x^{3/4}-0.05\right)\right)$	$0\le x_i\le 1$	4/1
F4: Equal Maxima	$sin^6(5\pi x)$	$0\le x_i\le 1$	0/5
F5: Himmelblau	$200-\left(x^2+y-11\right)^2-\left(x+y^2-7\right)^2$	$-6\le x,y\le 6$	0/4
F6: Modified Rastrigin	$-\sum_{i=1}^{n}[10+9\cos(2\pi k_i x_i)]$	$0\le x_i\le 1$	0/12
F7: Vincent	$\frac{1}{D}\sum_{i=1}^{D}\sin(10\log(x_i))$	$0.25\le x\le 10$	0/36
F8: Shubert	$\left(\sum_{i=1}^{5}icos((i+1)x_1+i)\right)\left(\sum_{i=1}^{5}icos((i+1)x_2+i)\right)$	$-5\le x_1\le 5$ $-5\le x_2\le 5$	742/18
F9: Composition function 1	$f_1:f_2$: Grienwank's function, $f_3:f_4$: Weierstrass function, and $f_5:f_6$: Sphere function.$\lambda=[1,1,8,8,1/5,1/5]$ $\sum_{i=1}^{6}\omega_i(f_i((x-o_i)/\lambda_i.M_i)+bias_i)+f_{bias}$	$-5\le x\le 5$	119/6
F10: Composition function 2	$f_1:f_2$: Rastrigin's function, $f_3:f_4$: Weierstrass function, $f_5:f_6$: Griewank's function, and $f_7:f_8$: Sphere function.$\lambda=[1,1,10,10,1/10,1/10,1/7,1/7]$ $\sum_{i=1}^{8}\omega_i(f_i((x-o_i)/\lambda_i.M_i)+bias_i)+f_{bias}$	$-5\le x\le 5$	264/8
F11: Five-Uneven-Peak Trap	$700-\sum_{i=1}^{D}t_i+200Dt_i=\begin{cases}-200+x_i^2,x_i<0\\-80(2.5-x_i),0\le x_i<2.5\\-64(x_i-2.5),2.5\le x_i<5\\-64(7.5-x_i),5\le x_i<7.5\\-28(x_i-7.5),7.5\le x_i<12.5\\-28(17.5-x_i),12.5\le x_i<17.5\\-32(x_i-17.5),17.5\le x_i<22.5\\-32(27.5-x_i),22.5\le x_i<27.5\\-80(x_i-27.5),27.5\le x_i\le 30\\-200+(x_i-30)^2,x_i>30\end{cases}$	$-100\le x_1\le -45$ $30\le x_2\le 80$	21/4
F12: Expanded Equal Maxima	$307-\sum_{i=1}^{D}t_i+Dt_i=\begin{cases}y_i^2,y_i<0ory_i>1\\-sin^6(5\pi y_i)0\le y_i\le 1\end{cases}$	$10\le y_1\le 50$ $-40\le y_2\le 10$	0/25
F13: Expanded Uneven Maxima	$508-\sum_{i=1}^{D}t_i-Dt_i=\begin{cases}y_i^2,y_i<0ory_i>1\\-sin^6\left(5\pi(y_i\frac{3}{4}-0.05)\right)0\le y_i\le 1\end{cases}$	$-40\le y_1\le 0$ $-70\le y_2\le -30$	0/25
F14: Modified Vincent	$805-\frac{1}{D}\sum_{i=1}^{D}t_i+1.0$ $t_i=\begin{cases}\sin(10\log(y_i)),0.25\le y_i\le 10\\(0.25-y_i)^2+\sin(10\log(2.5)),y_i<0.25\\(y_i-10)^2+\sin(10\log(10))y_i>10\end{cases}$	$20\le y_1\le 80$ $40\le y_2\le 100$	0/36

To ensure an unbiased comparison and prevent randomness, each algorithm is executed 25 times. Additionally, the precise value of 0.1 is consistently applied to the level of accuracy across all experiments. As termination criteria, the maximum number of function evaluations has been applied. 100,000 is assigned to F1, F6, and F7, and 10,000 is assigned to F2–F4. Additionally, the upper limit for the number of function evaluations is 50,000 for F5 and 200,000 for F8–F14.

The detailed performance criteria comparison results on F1–F14 between MBRO and other multimodal algorithms are shown in Table 2, 3, 4, 5, and 6. The top results and positions are highlighted for ease of reference. When we look at the success rate performance of the proposed algorithm, it outperforms other algorithms, as seen in

Table 2 Experimental results in success rate (↑) on problems F1–F14

Function/ Algorithm	MBRO	CFA	EPSO	FERPSO	LSPSO	NGSA	MFPA	MAMO
F1	0	0	0	0	0	0	0	0
F2	**0.8**	0	0.6	0	0	0	0	**0.8**
F3	**1**	0	0	1	1	**1**	0	**1**
F4	**1**	0	0	1	1	**1**	0	**1**
F5	**1**	**1**	**1**	0	1	0	0	**1**
F6	**1**	0	0	0	0	1	0	0,2
F7	0	0	0	0	0	0	0	0
F8	0	0	0	0	0	0	0	0
F9	0	0	0	0	0	0	0	0
F10	0	0	0	0	0	0	0	0
F11	**1**	0	0	0	0	0	0	0.4
F12	**1**	**1**	0.2	1	0.6	0	0	**1**
F13	**1**	**1**	**1**	0.8	**1**	0	0	**1**
F14	0	0	0	0.2	0.2	0	0	0

The best results are given in bold

Table 3 Experimental results in Success Performance (↓) on problems F1–F14

Function/ Algorithm	MBRO	CFA	EPSO	FERPSO	LSPSO	NGSA	MFPA	MAMO
F1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
F2	**12,500**	NaN	16,666.67	NaN	NaN	NaN	NaN	**12,500**
F3	**10,000**	NaN	NaN	**10,000**	**10,000**	**10,000**	NaN	**10,000**
F4	**10,000**	NaN	NaN	**10,000**	**10,000**	**10,000**	NaN	**10,000**
F5	**50,000**	**50,000**	**50,000**	NaN	**50,000**	NaN	NaN	**50,000**
F6	100,000	NaN	NaN	NaN	NaN	10,000	NaN	50,000
F7	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
F8	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
F9	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
F10	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
F11	**200,000**	NaN	NaN	NaN	NaN	NaN	NaN	500,000
F12	**200,000**	**200,000**	1,000,000	250,000	333,333.3	NaN	NaN	200,000
F13	**200,000**	**200,000**	**200,000**	250,000	**200,000**	NaN	NaN	**200,000**
F14	NaN	NaN	NaN	1,000,000	1,000,000	NaN	NaN	NaN

NaN Not a number

The best results are given in bold

Table 2. Additionally, it is the only algorithm that can find local/global solutions in the F11 function. The algorithms were able to find all the optimal points in F1, F7, F8, and F9 and F14 even once. Based on this table, it appears that MBRO has a consistent track record of success across multiple functions. Accordingly, Success Performance cannot be determined in these cases. Table 4 summarizes the average optima found for functions F1 to F14. MBROA is the only algorithm that finds the maximum number of optimal points in the F1, F2, F7, F8, F9, F10, and F11 test functions. Although it doesn't rank first in the F14 function, it successfully locates 31.6 out of 36 local/global points.

The proposed algorithm demonstrates the best performance in terms of MPR among 14 test functions used, achieving the highest results in 12 of them. (Table 5) However, in the case of functions F3, F4, F5, F7, F12, and F13, some of the compared algorithms can achieve a similar level of success as MBRO and can perform equally well as the proposed method.

MBRO's performance on functions F12, F13, and F14, where it finds all or nearly all optima (See Table 4), suggests it maintains good population diversity across runs, as it is not getting stuck on a single peak or subset of peaks. However, the true test of population diversity would be how MBRO performs on functions with a large number of local optima;

Table 4 Experimental results in Average Optima Found (↑) on problems F1–F14

Function/ Algorithm	MBRO	CFA	EPSO	FERPSO	LSPSO	NGSA	MFPA	MAMO
F1	**116.4**	56.4	49	31.8	65.6	33.4	0.4	101.2
F2	**5.8**	2.2	5.2	3.6	4.4	2.2	0	5.6
F3	**5**	1	1	**5**	**5**	**5**	1.6	**5**
F4	**5**	1.6	1	**5**	**5**	**5**	2	**5**
F5	**4**	**4**	**4**	2.8	**4**	0.8	0	**4**
F6	**12**	3.6	1.4	3.2	3.6	**12**	0.6	10.2
F7	**24**	13.2	13	11.8	21.4	7	0	22.2
F8	**595.4**	108	138.6	185	389.2	100.6	0	450
F9	**76**	10.2	17	24.2	49.4	26.2	0.2	56.6
F10	**224**	21.2	51.4	106.4	177.2	93	0	199.4
F11	**25**	14.4	18.8	18	22.4	0.6	0	24.2
F12	**25**	**25**	24	**25**	24.6	0.8	0	**25**
F13	**25**	**25**	**25**	24.8	**25**	1	0	**25**
F14	31.6	27.4	27.6	**35**	34	0.4	0	32.6

The best results are given in bold

Table 5 Experimental results in Maximum Peak Ratio (↑) on problems F1–F14

Function/ Algorithm	MBRO	CFA	EPSO	FERPSO	LSPSO	NGSA	MFPA	MAMO
F1	**0.98**	0.59	0.48	0.68	0.66	0.72	0	0.88
F2	**0.99**	0.42	0.85	0.68	0.87	0.21	0	0.97
F3	**0.99**	0.29	0.29	**0.99**	**0.99**	**0.99**	0.05	**0.99**
F4	**0.99**	0.32	0.20	**0.99**	**0.99**	**0.99**	0.1	**0.99**
F5	**0.99**	**0.99**	**0.99**	0.57	**0.99**	0.23	0	0.83
F6	**0.99**	0.28	0.15	0.62	0.49	**0.99**	0.8	0.44
F7	**0.89**	0.32	0.36	0.23	0.38	0.15	0	**0.89**
F8	**0.85**	0.19	0.23	0.41	0.54	0.35	0	0.61
F9	**0.70**	0.16	0.48	0.31	0.59	0.55	0.07	0.34
F10	**0.95**	0.10	0.23	0.81	0.74	0.66	0.14	**0.88**
F11	0.66	0.67	**0.87**	0.78	0.82	0.08	0	0.72
F12	**0.99**	0.89	0.96	0.99	0.99	0.14	0	**0.99**
F13	**0.99**	0.99	0.97	0.99	0.99	0.1	0	**0.99**
F14	0.85	0.85	0.82	0.99	0.84	0.03	0	**0.87**

The best results are given in bold

Table 6 Average ranks of algorithms

Function/algorithm	MBRO	CFA	EPSO	FERPSO	LSPSO	NGSA	MFPA	MAMO
Success rate	**1**	5	6	4	3	6	8	2
Success performance	**1**	4	6	5	3	6	8	**1**
Avg. optima found	**1**	5	6	4	3	7	8	2
Max. peak ratio	**1**	7	6	4	3	5	8	2
Average rank	1	5.33	6	4.33	3	6	8	1.66
Overall rank	**1**	5	6	4	3	6	8	2

The best results are given in bold

F8 might be such a case, and MBRO does exceptionally well here. Across all listed functions, MBRO demonstrated superior performance by identifying a greater number of optima compared to the other algorithms. This suggests that MBRO is likely maintaining a diverse population, allowing it to explore various regions of the search space effectively.

In the Table 4, MBRO seems to perform exceptionally well on F1 and F8, which suggests that it might be scalable to problems with a large number of optima. On F1, it found an average of 116.4 optima, and on F8, an impressive 595.4 optima. These functions might represent larger or more complex search spaces, although without additional context on the nature of these functions, this is speculative. If F1 and F8 are indeed more complex, MBRO's performance here could indicate good scalability.

MBRO's high average optima found in several functions (like F1, F8, F10) suggest it is efficient in these instances, especially when compared to other algorithms. However, on some functions like F2, F3, F4, and F5, its efficiency seems to be on par with the other algorithms, as they all find a similar number of optima. Without data on the resources used by MBRO to achieve these results, we can't fully assess its efficiency, but the high number of optima found suggests it could be quite efficient.

As seen in Table 6, the proposed algorithm ranks first among all performance criteria. While sharing the first place in terms of success performance with MAMO, in the overall performance criteria, MAMO ranks second, following them, FERPSO is in the fourth position, CFA is in the fifth position, and EPSO and NGSA share the sixth position, with MFPA placed in the last position.

In summary, the comprehensive analysis of the performance criteria demonstrates the effectiveness of the proposed MBROA. It outperforms other multimodal algorithms in the majority of the test functions, achieving remarkable success in terms of success rate, optimal point findings, and maximum peak ratio (MPR). Specifically, MBRO excels in functions F1, F2, F7, F8, F9, F10, and F11 by locating the maximum number of optimal points, demonstrating its ability to find both local and global solutions. Furthermore, it exhibits superior performance in MPR, surpassing other algorithms in 12 out of the 14 test functions.

While some compared algorithms exhibit competitive performance in specific functions, MBRO consistently ranks first overall. It shares the top position in success performance with MAMO but takes the lead when considering the broader performance criteria. MAMO follows as the second-ranking algorithm, with FERPSO, CFA, EPSO, NGSA, and MFPA following in descending order. These results highlight the robustness and versatility of MBRO, positioning it as a highly effective choice for multimodal optimization tasks. The achievements presented in this study offer promising insights into the practical applications of MBROA in various fields, emphasizing its potential for solving complex optimization challenges.

MBRO has had perfect success rates (1) on some functions, like F3, F4, F5, F6, F11, F12, and F13, which means it can consistently find all existing optima in those cases. However, like other algorithms (a,b), it has failed miserably on other functions, like F1, F7, F8, F9, F10, and F14. The success rate was 0 because it could never find all 121 optimums, even though it found an average of 116.4 out of 121 in function 1. This inconsistency could point to limitations in handling certain types of problem landscapes or objective functions.

To overcome challenges, MBRO's exploration strategies could be strengthened by incorporating adaptive mutation rates, crossover strategies, or diversity preservation techniques. Hybridization with other optimization techniques could compensate for weaknesses, such as combining MBRO with algorithms that performed well on functions where MBRO failed. Fine-tuning the algorithm's parameters, such as population size and selection pressure, could also improve success rates. Customizing the algorithm to better suit the characteristics of problems where it currently fails could also be a way forward. This could involve designing custom operators or heuristics for specific landscapes.

5 Conclusion

The presented research has addressed the intricate and pressing challenges of multimodal optimization, where the identification of multiple local and global optima is paramount. The contribution of this study lies in the development of the MBRO approach, which sets itself apart by its ability to navigate these complex landscapes without necessitating prior knowledge of the problem space and by eliminating the need to predefine the niching parameter. MBRO compares solutions based on their largest electrostatic force, rather than the nearest neighbor. If a solution's damage level doesn't exceed a predetermined threshold, a standard "run-and-tumble" motion is executed to improve local search accuracy. If the damage level exceeds a threshold, the solution moves towards the winning solution chosen by Coulomb's law. The results of our experiments, which compared MBRO against seven established multimodal optimization algorithms across 14 diverse benchmark problems, are highly promising. MBRO consistently demonstrated its capacity to locate most, if not all, local and global optima, surpassing its competition in terms of performance. This performance, combined with the algorithm's adaptability to diverse problem spaces, makes it a valuable addition to the toolkit of researchers and practitioners seeking efficient solutions for multimodal optimization tasks. The study highlights the significant performance and applicability of MBRO, a promising avenue for future research and practical implementation. It predicts that the findings will inspire further exploration and application of MBRO in various domains, contributing to more effective problem-solving strategies.

Author contributions TA provided core concepts, KDC and TA drafted the manuscript and carried out implementations and

simulations for this manuscript. OB proofread the manuscript and approved the final manuscript.

Funding Open access funding provided by the Scientific and Technological Research Council of Türkiye (TÜBİTAK). The authors have not disclosed any funding.

Data availability Enquiries about data availability should be directed to the authors.

Declarations

Conflict of interest Conflict of interest the authors declare no conflict of interest in this study.

References

1. Li, X.: A multimodal particle swarm optimizer based on fitness Euclidean-distance ratio. In: Proceedings of GECCO 2007: Genetic and Evolutionary Computation Conference, pp. 78–85 (2007). https://doi.org/10.1145/1276958.1276970
2. Yang, Q., et al.: Adaptive multimodal continuous ant colony optimization. IEEE Trans. Evol. Comput. **21**(2), 191–205 (2017). https://doi.org/10.1109/TEVC.2016.2591064
3. Goldberg, D.E., Richardson, J.: Genetic algorithms with sharing for multimodal function optimization. In: Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms: July 28–31, 1987 at the Massachusetts. Institute of Technology, Cambridge (1987)
4. Goldberg, D.E., Goldberg, D.E., Wang, L.: Adaptive niching via coevolutionary sharing. In: Genetic Algorithms and Evolution Strategy in Engineering and Computer Science, Chapter 2, pp. 21–38 (1997)
5. Lin, C.Y., Wu, W.H.: Niche identification techniques in multimodal genetic search with sharing scheme. Adv. Eng. Softw. **33**(11–12), 779–791 (2002). https://doi.org/10.1016/S0965-9978(02)00045-5
6. Miller, B.L. Shaw, M.J.: Genetic algorithms with dynamic niche sharing for multimodal function optimization. In: Proceedings of the IEEE Conference on Evolutionary Computation, IEEE, pp. 786–791 (1996). https://doi.org/10.1109/icec.1996.542701
7. Sareni, B., Krähenbühl, L.: Fitness sharing and niching methods revisited. IEEE Trans. Evol. Comput. **2**(3), 97–106 (1998). https://doi.org/10.1109/4235.735432
8. Yin, X., Germay, N.: A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization. In: Artificial Neural Nets and Genetic Algorithms, pp. 450–457. Springer, Vienna (1993). https://doi.org/10.1007/978-3-7091-7533-0_65
9. De Jong, K.A.: Analysis of the Behavior of a Class of Genetic Adaptive Systems. University of Michigan, Ann Arbor (1975)
10. Sw, M.: Crowding and preselection revisited. Parallel Probl. Solving Nat **2**(2), 27–36 (1992)
11. Petrowski, A.: Clearing procedure as a niching method for genetic algorithms. In: Proceedings of the IEEE Conference on Evolutionary Computation, pp. 798–803. IEEE (1996). https://doi.org/10.1109/icec.1996.542703.
12. Li, J.P., Balazs, M.E., Parks, G.T., Clarkson, P.J.: A species conserving genetic algorithm for multimodal function optimization. Evol. Comput. **10**(3), 207–234 (2002). https://doi.org/10.1162/106365602760234081
13. Li, J.-P., Wood, A.S.: An adaptive species conservation genetic algorithm for multimodal optimization. Int. J. Numer. Methods Eng. **79**(13), 1633–1661 (2009). https://doi.org/10.1002/nme.2621
14. De Jong, K.A.: An Analysis of the Behavior of a Class of Genetic Adaptive Systems (Doc- 1991). University of Michigan, Ann Arbor (1975)
15. Mengshoel, O.J., Goldberg, D.E.: Probabilistic crowding: deterministic crowding with probabilistic replacement, pp. 409–416 (1999)
16. Holland, J.: Adaptation in Natural and Artificial Systems: An Introductory Analysis with Application to Biology, Control and Artificial Intelligence. MIT Press, Cambridge (1975)
17. Pétrowski, A.: An efficient hierarchical clustering technique for speciation. Evolution. Technical report, Institute National des Telecommunications, Evry, France, Technique Report (2007)
18. Barrera, J., Coello, C.A.C.: A particle swarm optimization method for multimodal optimization based on electrostatic interaction. In: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 622–632 (2009). https://doi.org/10.1007/978-3-642-05258-3_55
19. Qu, B.Y., Liang, J.J., Suganthan, P.N.: Niching particle swarm optimization with local search for multi-modal optimization. Inf. Sci. (N Y) **197**, 131–143 (2012). https://doi.org/10.1016/j.ins.2012.02.011
20. Rahkar-Farshi, T., Behjat-Jamal, S.: A multimodal firefly optimization algorithm based on Coulomb's law. Int. J. Adv. Comput. Sci. Appl. (2016). https://doi.org/10.14569/ijacsa.2016.070521
21. Gálvez, J., Cuevas, E., Avalos, O.: Flower Pollination algorithm for multimodal optimization. Int. J. Comput. Intell. Syst. **10**(1), 627–646 (2017). https://doi.org/10.2991/ijcis.2017.10.1.42
22. Farshi, T.R.: A memetic animal migration optimizer for multimodal optimization. Evolv. Syst. (2021). https://doi.org/10.1007/s12530-021-09368-3
23. Farshi, T.R., Orujpour, M.: A multi-modal bacterial foraging optimization algorithm. J. Ambient. Intell. Humaniz. Comput.

12(11), 10035–10049 (2021). https://doi.org/10.1007/s12652-020-02755-9
24. Orujpour, M., Feizi-Derakhshi, M.R., Akan, T.: A multimodal butterfly optimization using fitness-distance balance. Soft. Comput. (2023). https://doi.org/10.1007/s00500-023-09074-z
25. Farshi, T.R.: Battle Royale optimization algorithm. Neural Comput. Appl. **33**(4), 1139–1157 (2021). https://doi.org/10.1007/s00521-020-05004-4
26. Li, X., Engelbrecht, A., Epitropakis, M.G.: Benchmark Functions for CEC'2013 Special Session and Competition on Niching Methods for Multimodal Function Optimization. RMIT University, Melbourne (2013)
27. Qu, B.Y., Liang, J.J., Suganthan, P.N., Chen, Q.: Problem definitions and evaluation criteria for the CEC 2015 competition on single objective multi-niche optimization. http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2015/CEC2015.htm

K. Dilşad Çiçek is currently pursuing the Ph.D. degree with the Graduate School of Science and Engineering, Altinbas University, Istanbul, Turkey.

Taymaz Akan is a postdoctoral fellow in the School of Medicine at Louisiana State University, USA (Shreveport). He received his Ph.D. from the Department of Computer Engineering at Gazi University, Ankara, Turkey, in 2019. He was the Director of the Distance Education Research and Application Center and also the Vice Dean of the Faculty of Engineering at Topkapi University.

Oguz Bayat received the B.S. degree from Istanbul Technical University, Istanbul, Turkey, in 2000, the M.S. degree from the University of Hartford, CT, USA, in 2002, and the Ph.D. degree from Northeastern University, Boston, MA, USA, in 2006, all in electrical engineering. Before attending the Communication and Digital Signal Processing Research Center at Northeastern University in 2002, he served as an Adjunct Faculty Member at the Department of Electrical and Computer Engineering, University of Hartford. He is currently working as a Technical Leader/Manager on the Nortel project at Airvana Inc., Boston. He has been involved in the research and development of CDMA 1xEV-DO Rev 0, Rev A, and Rev B radio node and radio node controller software products since 2005. Since 2011, he has been an Associate Professor with the Department of Electrical and Electronics Engineering, Altinbas University. His current research interests include wireless communications, digital signal processing, channel coding, channel estimation, and equalization techniques, in particular multilevel coding, turbo coding, and equalization for wireless communications.